

1/17

FIG. 1

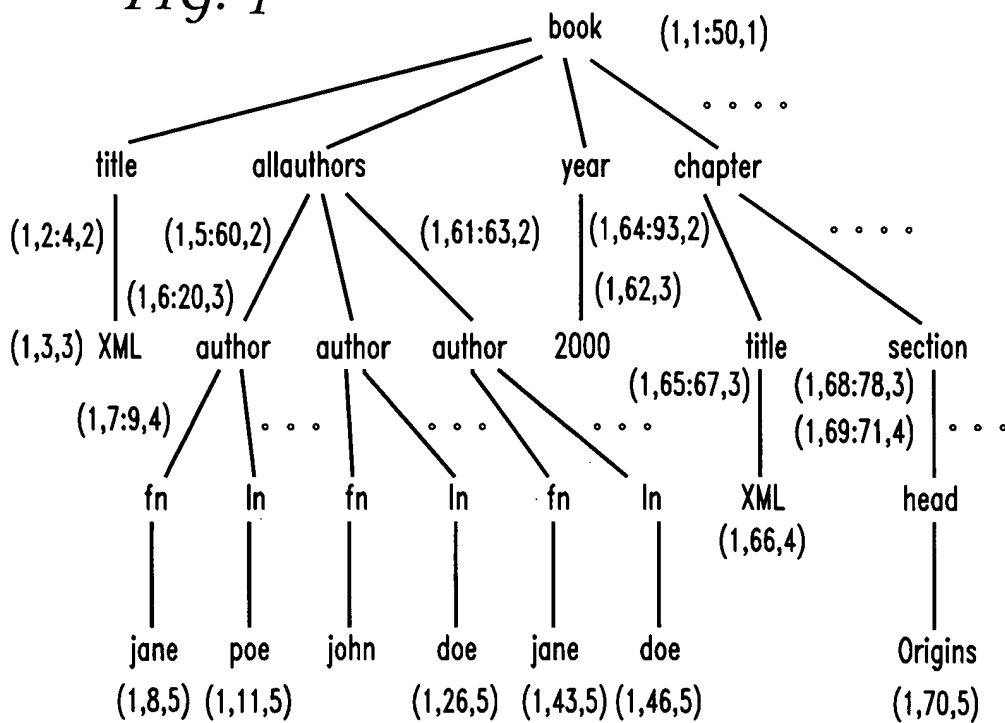


FIG. 2A

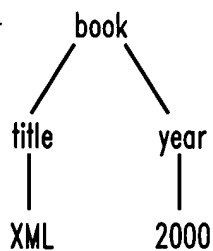
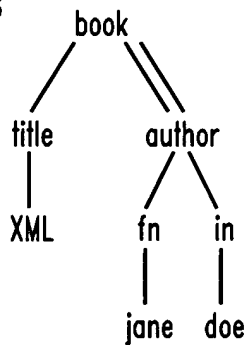


FIG. 2B



2/17

FIG. 3A

A_1
 $|$
 B_1
 $|$
 A_2
 $|$
 B_2
 $|$
 C_1
 DATA

FIG. 3B

A
 $||$
 B
 $||$
 C
 QUERY

FIG. 3C

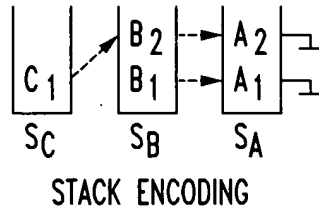


FIG. 3D

A_1 B_1 C_1
 A_1 B_2 C_1
 A_2 B_2 C_1
 QUERY RESULTS

FIG. 4

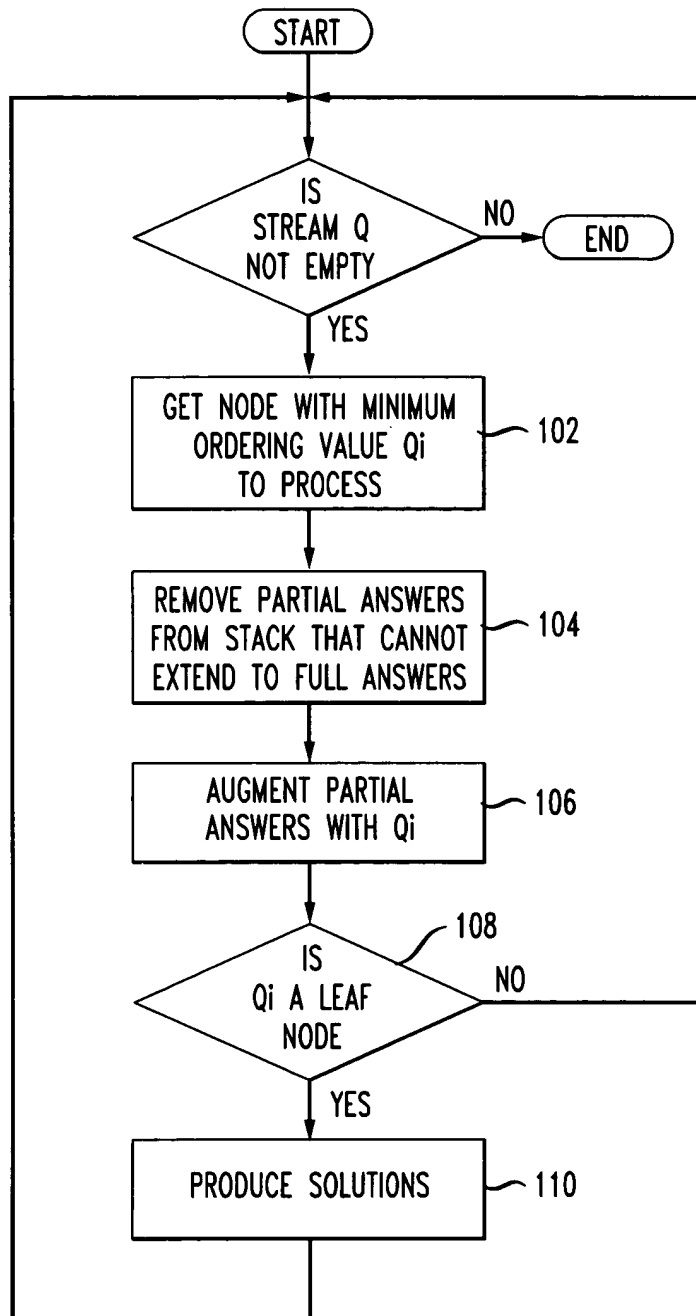
```

Algorithm PathStack( $q$ )
01  while  $\neg \text{end}(q)$ 
02     $q_{\min} = \text{getMinSource}(q)$ 
03    for  $q_i$  in subtreeNodes( $q$ ) // clean stacks
04      while ( $\neg \text{empty}(S_{q_i}) \wedge \text{topB}(S_{q_i}) < \text{nextL}(T_{q_{\min}})$ )
05        pop( $S_{q_i}$ )
06      moveStreamToStack( $T_{q_{\min}}$ ,  $S_{q_{\min}}$ , pointer to
                        top( $S_{\text{parent}(q_{\min})}$ ))
07    if (isleaf( $q_{\min}$ ))
08      showSolutions( $S_{q_{\min}}$ , 1)
09      pop( $S_{q_{\min}}$ )
Function end( $q$ )
  return  $\forall q_i \in \text{subtreeNodes}(q): \text{isLeaf}(q_i) \Rightarrow \text{eof}(T_{q_i})$ 
Function getMinSource( $q$ )
  return  $q_i \in \text{subtreeNodes}(q)$  such that nextL( $T_{q_i}$ )
  is minimal
Procedure moveStreamToStack( $T_q, S_q, p$ )
01  push( $S_q, (\text{next}(T_q), p)$ )
02  advance( $T_q$ )
  
```

PathStack

3/17

FIG. 4A



4/17

FIG. 5

```

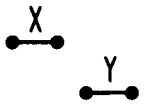
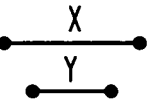
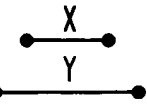
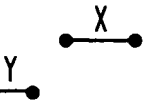
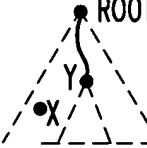
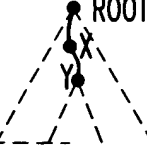
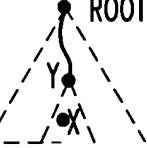
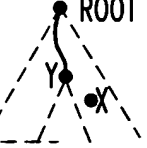
Procedure showSolutions(SN,SP)
// Assume, for simplicity, that the stack of the query
// nodes from the root to the current leaf node we
// are interested in can be accessed as S[1],...,S[n].
// Also assume that we have a global array index[1..n]
// of pointers to the stack elements.
// index[i] represents the position in the i'th stack that
// we are interested in for the current solution, where
// the bottom of each stack has position 1

// Mark we are interested in position SP of stack SN.
01 index[SN] = SP
02 if (SN == 1) // we are in the root
03   // output solutions from the stacks
04   output (S[n].index[n],...,S[1].index[1])
05 else // recursive call
06   for i = 1 to S[SN].index[SN].pointer_to_parent
07     showSolutions(SN - 1,i)

```

Procedure showSolutions

FIG. 6

	CASE 1	CASE 1	CASE 1	CASE 1
PROPERTY	$X.R < Y.L$	$X.L < Y.L$ $X.R > Y.R$	$X.L > Y.L$ $X.R < Y.R$	$X.L > Y.R$
SEGMENTS				
TREE				
CASES FOR PathStack AND TwigStack				

5/17

*FIG. 7*Algorithm PathMPMJ(q)

```

01 while ( $\neg \text{eof}(T_q) \wedge (\text{isRoot}(q) \vee$ 
       $\text{nextL}(q) < \text{nextR}(\text{parent}(q)))$ )
02   for ( $q_i \in \text{subtreeNodes}(q)$ ) // advance descendants
03     while ( $\text{nextL}(q_i) < \text{nextL}(\text{parent}(q_i))$ )
04       advance( $T_{q_i}$ )
05     PushMark( $T_{q_i}$ )
06   if ( $\text{isLeaf}(q)$ ) // solution in the streams' heads
07     outputSolution()
08   else PathMPMJ( $\text{child}(q)$ )
09   for ( $q_i \in \text{subtreeNodes}(q)$ ) // backtrack descendants
10     PopMark( $T_{q_i}$ )

```

PathMPMJ

6/17

FIG. 8

```

Algorithm TwigStack(q)
  // Phase 1
01  while ¬end(q)
02    qact = getNext(q)
03    if (¬isRoot(qact))
04      cleanStack(parent(qact), next(qact))
05    if (isRoot(qact) ∨ ¬empty(Sparent(qact)))
06      cleanStack(qact, next(qact))
07      moveStreamToStack(Tqact, pointer to
                                top(Sparent(qact)))
08    if (isLeaf(qact))
09      showSolutionWithBlocking(Sqact, 1)
10      pop(Sqact)
11    else advance(Tqact)
  // Phase 2
12  mergeAllPathSolutions()

Function getNext(q)
01  if (isLeaf(q)) return q
02  for qi in children(q)
03    ni = getNext(qi)
04    if (ni ≠ qi) return ni
05  nmin = minargni nextL(Tni)
06  nmax = maxrargni nextL(Tni)
07  while (nextR(Tq) < nextL(Tnmax))
08    advance(Tq)
09  if (nextL(Tq) < nextL(Tnmin)) return q
10  else return nmin

Procedure cleanStack(S, actL)
01  while (¬empty(S) ∧ (topR(S) < actL))
02    pop(S)

```

TwigStack

7/17

FIG. 8A

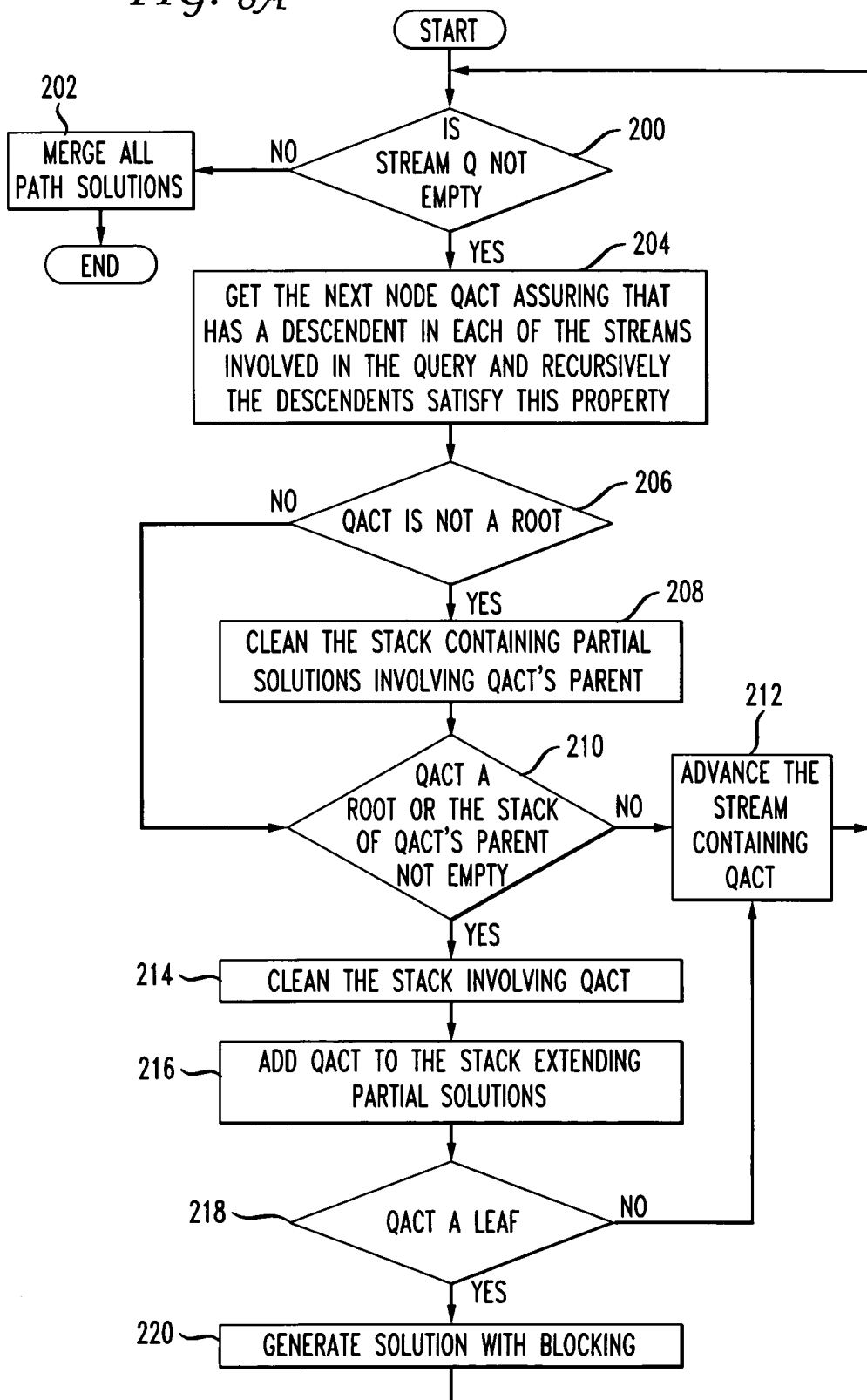


FIG. 9

8/17

```

Algorithm TwigStackXB(q)
01  while  $\neg \text{end}(q)$ 
02     $q_{\text{act}} = \text{getNext}(q)$ 
(03)  if ( $\text{isPlainValue}(Tq_{\text{act}})$ )
04    if ( $\neg \text{isRoot}(q_{\text{act}})$ )
05       $\text{cleanStack}(\text{parent}(q_{\text{act}}), \text{next}(q_{\text{act}}))$ 
06    if ( $\text{isRoot}(q_{\text{act}}) \vee \neg \text{empty}(\text{Sparent}(q_{\text{act}}))$ )
07       $\text{cleanStack}(q_{\text{act}}, \text{next}(q_{\text{act}}))$ 
08       $\text{moveStreamToStack}(Tq_{\text{act}}, \text{pointer to top}(\text{Sparent}(q_{\text{act}})))$ 
09    if ( $\text{isLeaf}(q_{\text{act}})$ )
10       $\text{showSolutionsWithBlocking}(Sq_{\text{act}}, 1)$ 
11       $\text{pop}(Sq_{\text{act}})$ 
12    else  $\text{advance}(Tq_{\text{act}})$ 
(13)  else if ( $\neg \text{isRoot}(q_{\text{act}}) \wedge \text{empty}(\text{Sparent}(q_{\text{act}})) \wedge \text{nextL}(T_{\text{parent}(q_{\text{act}})}) > \text{nextR}(Tq_{\text{act}})$ )
(14)   $\text{advance}(Tq_{\text{act}})$  // Not part of a solution
(15)  else // Might have a child in some solution
(16)   $\text{drillDown}(Tq_{\text{act}})$ 
      // Phase 2
17   $\text{mergeAllPathSolutions}()$ 

Function getNext(q)
01  if ( $\text{isLeaf}(q)$ ) return q
02  for  $q_i$  in  $\text{children}(q)$ 
03     $n_i = \text{getNext}(q_i)$ 
(04)  if ( $q_i \neq n_i \vee \neg \text{isPlainValue}(Tn_i)$ ) return  $n_i$ 
05   $n_{\text{min}} = \text{minarg}_{n_i} \text{nextL}(Tn_i)$ 
06   $n_{\text{max}} = \text{maxrarg}_{n_i} \text{nextL}(Tn_i)$ 
07  while ( $\text{nextR}(Tq) < \text{nextL}(Tn_{\text{max}})$ )
08     $\text{advance}(Tq)$ 
09  if ( $\text{nextL}(Tq) < \text{nextL}(Tn_{\text{min}})$ ) return q
10  else return  $n_{\text{min}}$ 

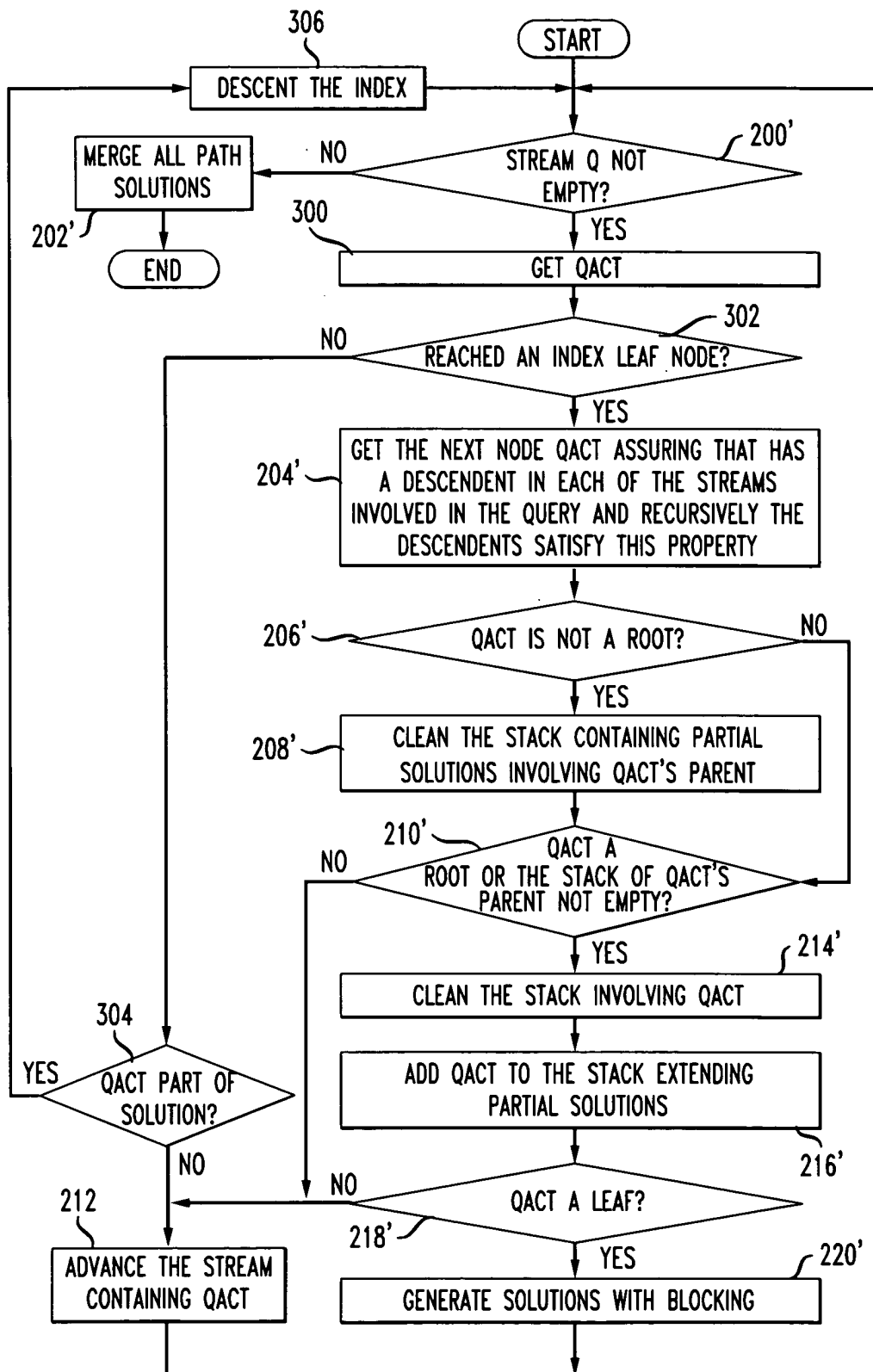
Procedure cleanStack(S, actL)
01  while ( $\neg \text{empty}(S) \wedge (\text{topR}(S) < \text{actL})$ )
02     $\text{pop}(S)$ 

```

TwigStack

FIG. 9A

9/17



10/17

FIG. 10

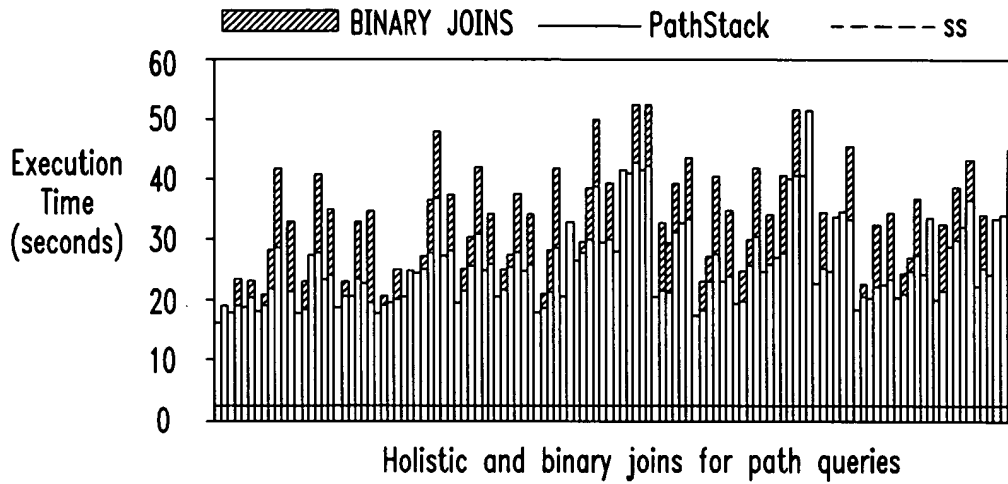
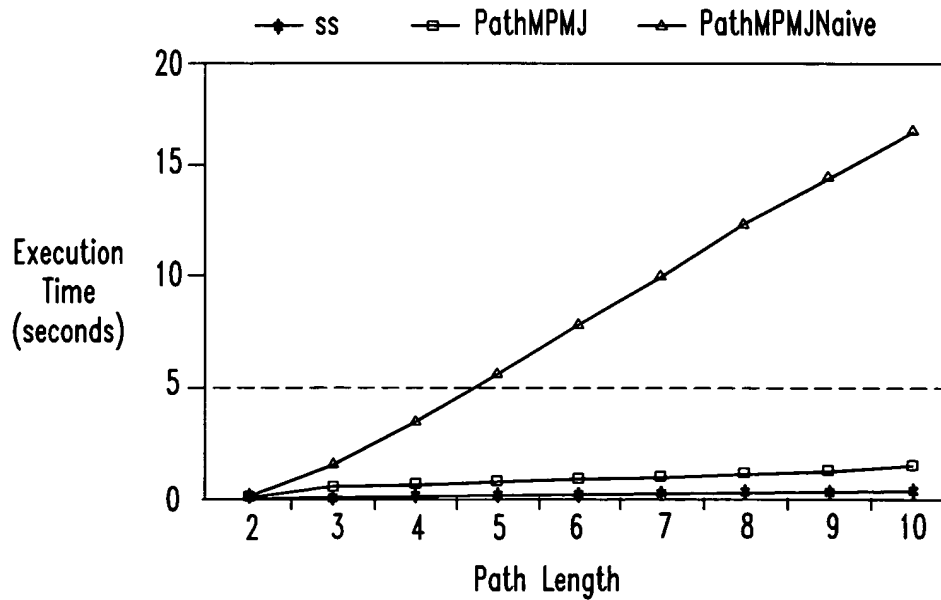


FIG. 11



11/17

FIG. 12A

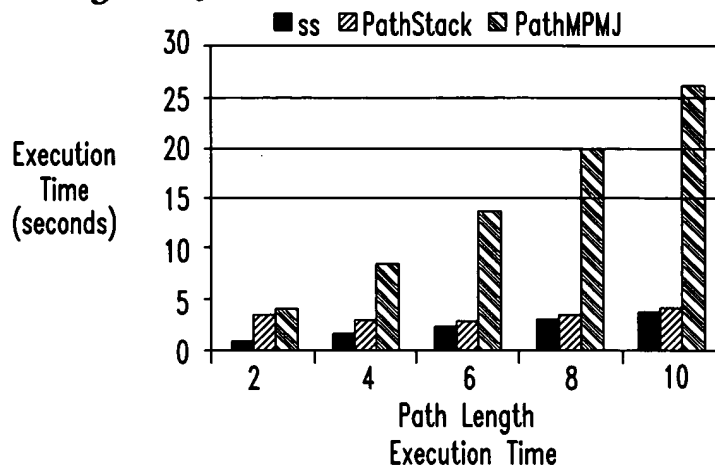
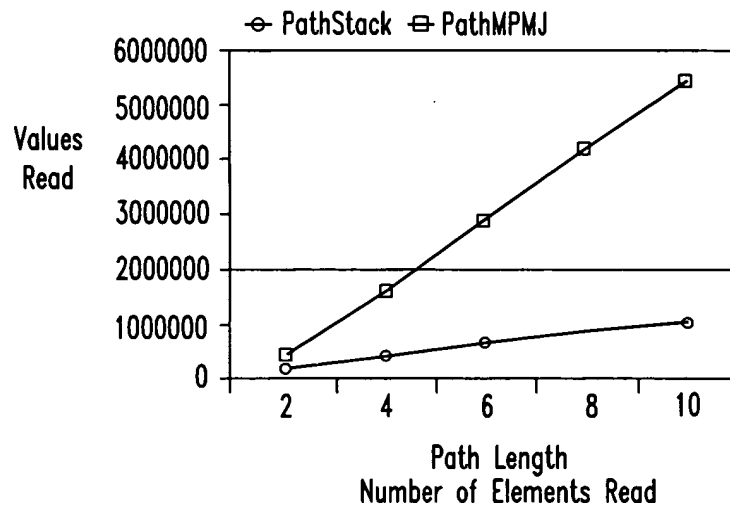


FIG. 12B



12/17

FIG. 13A

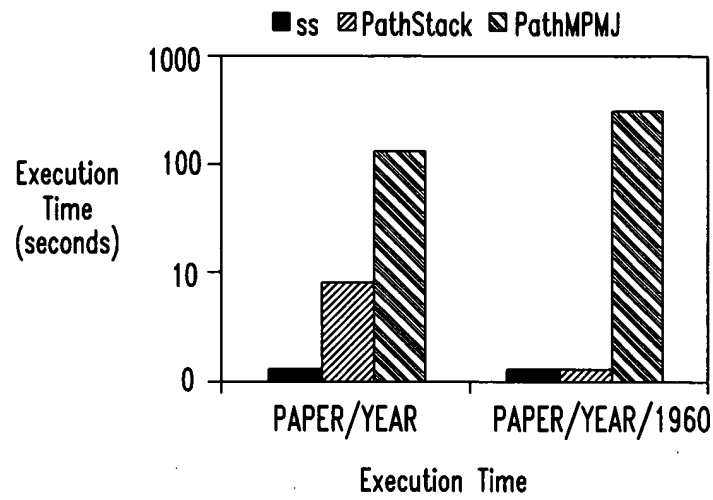
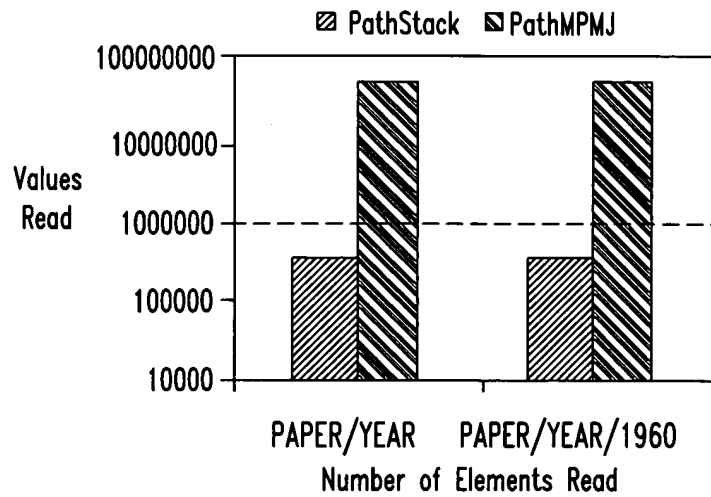
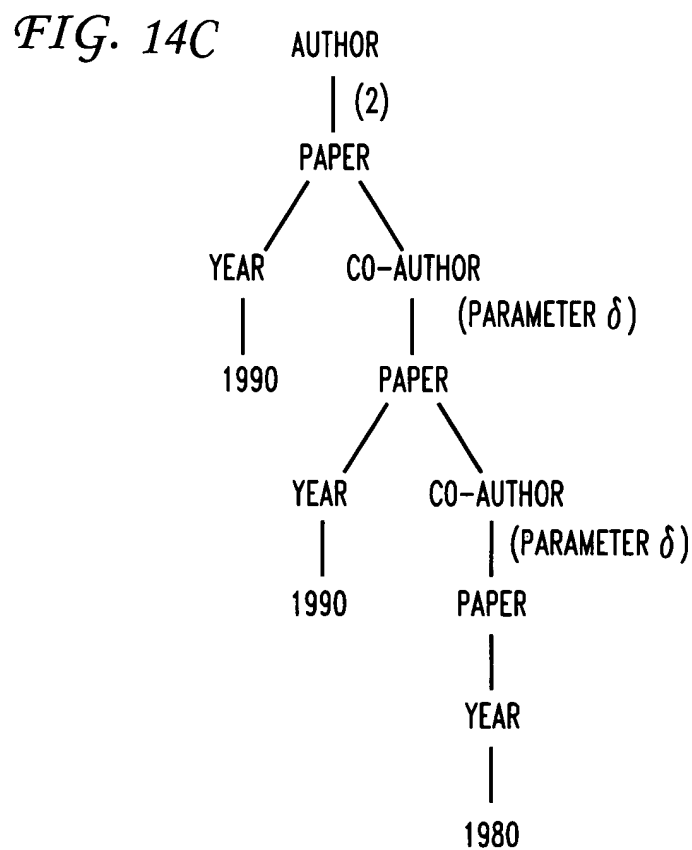
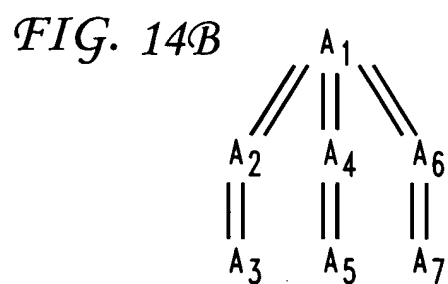
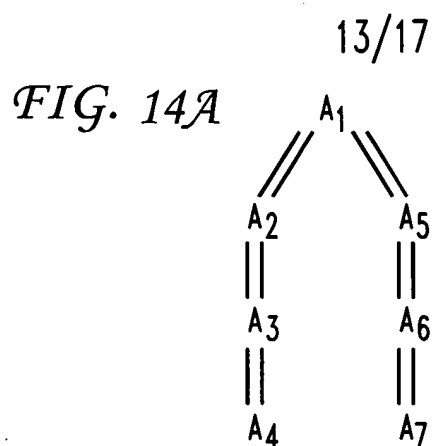


FIG. 13B





14/17

FIG. 15A

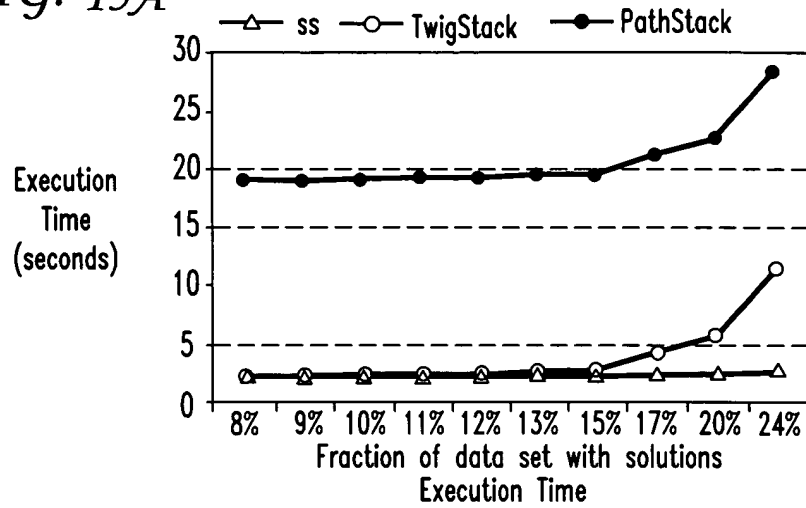


FIG. 15B

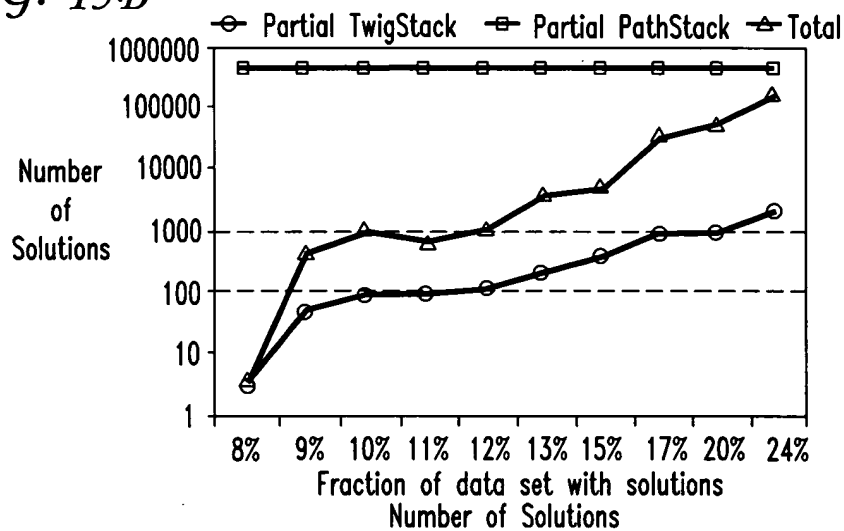
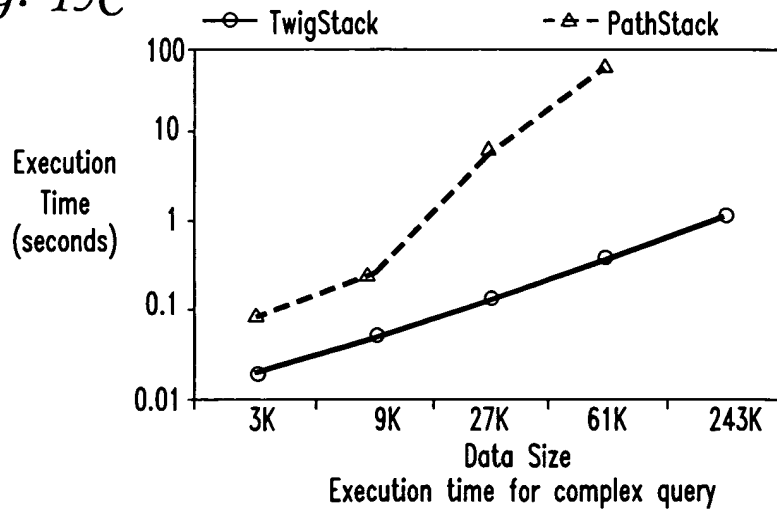


FIG. 15C



15/17

FIG. 16A

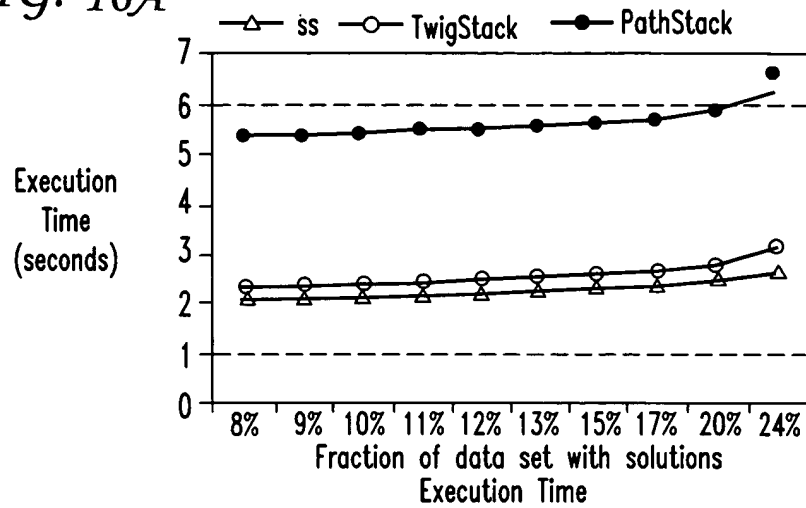


FIG. 16B

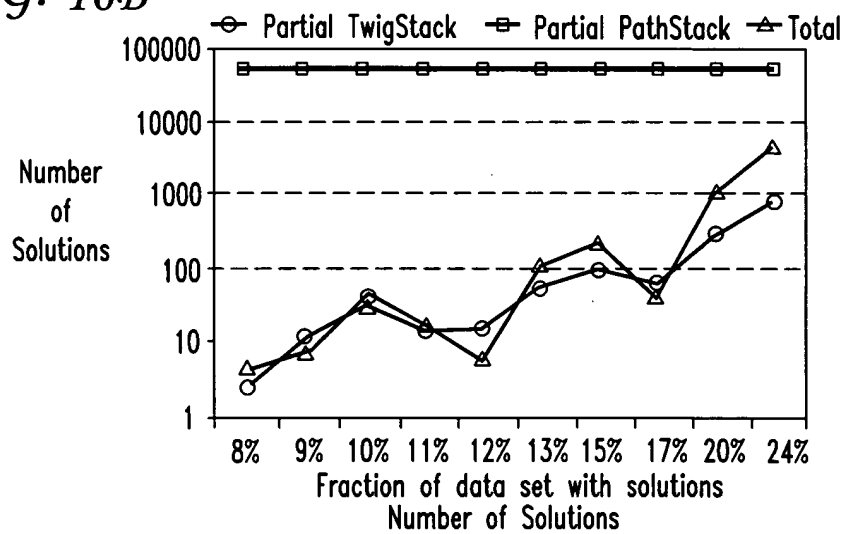
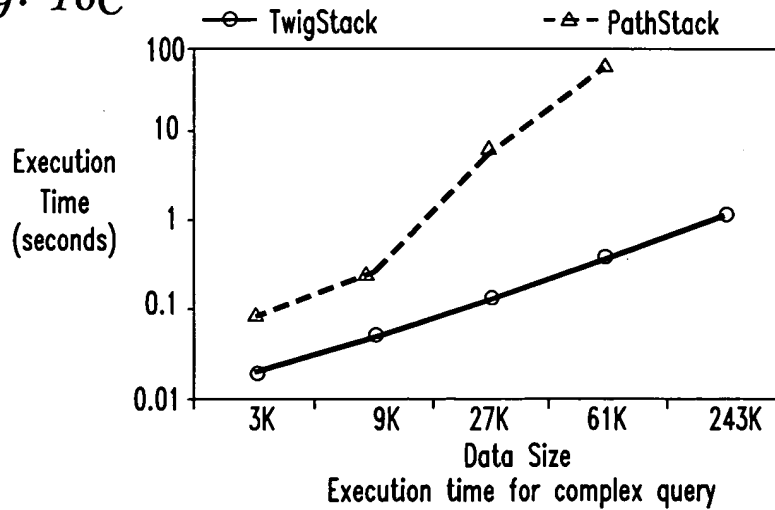


FIG. 16C



16/17

FIG. 17A

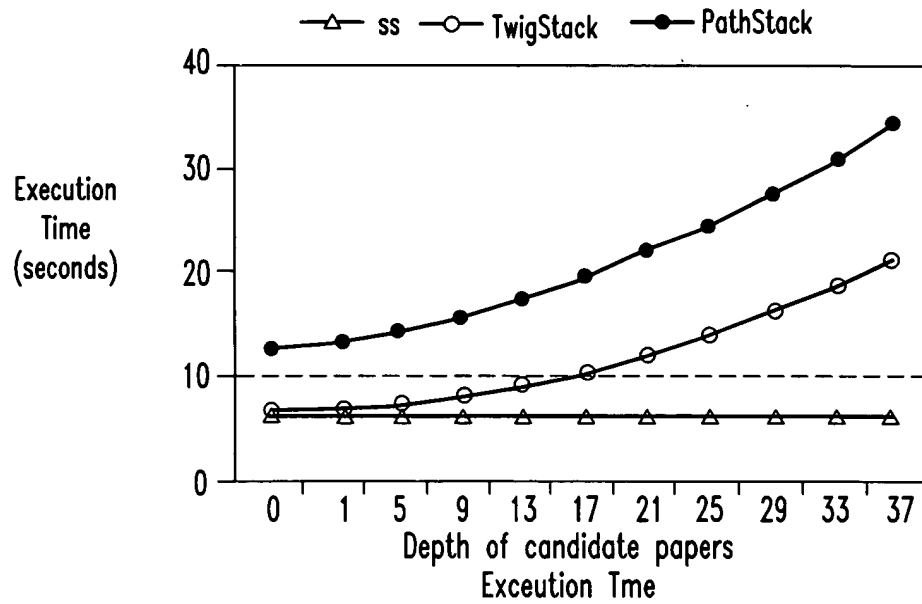
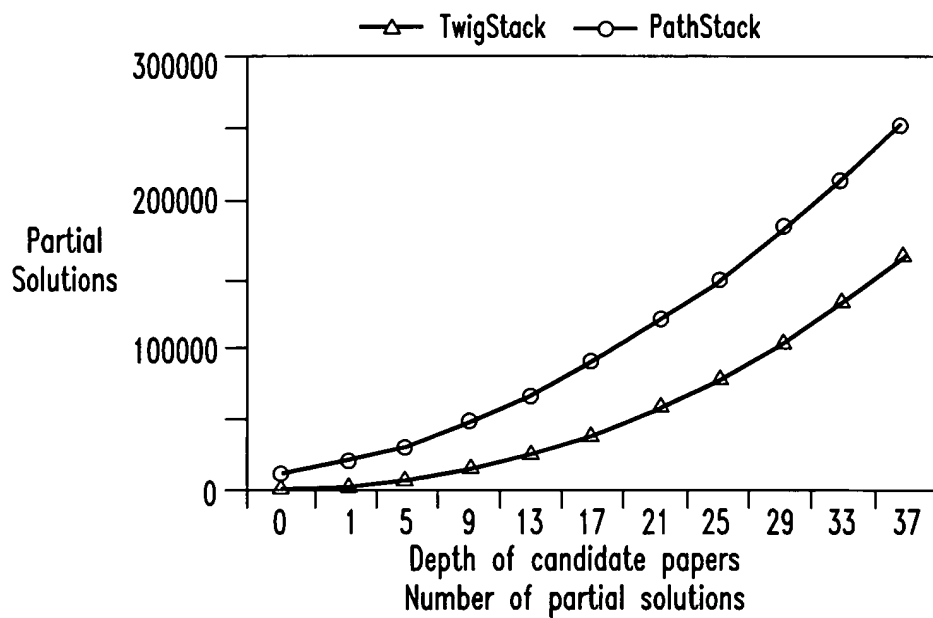


FIG. 17B



17/17

FIG. 18A

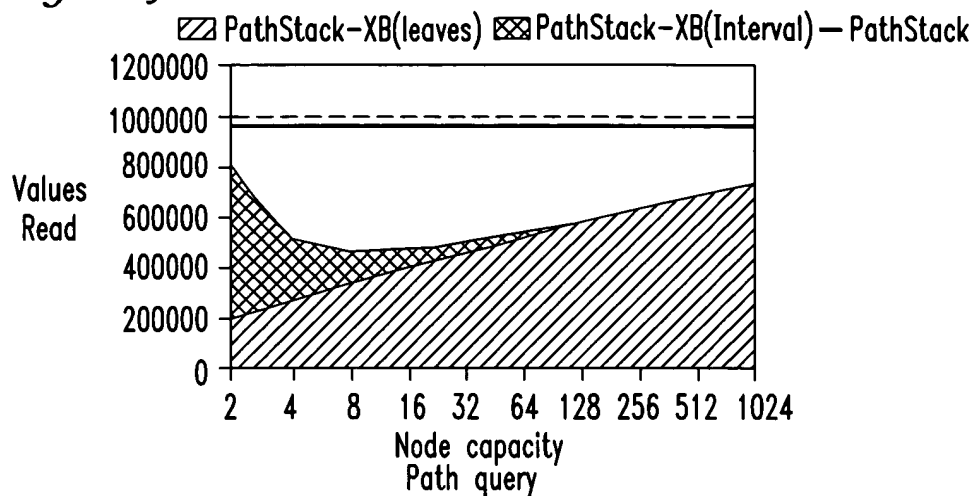


FIG. 18B

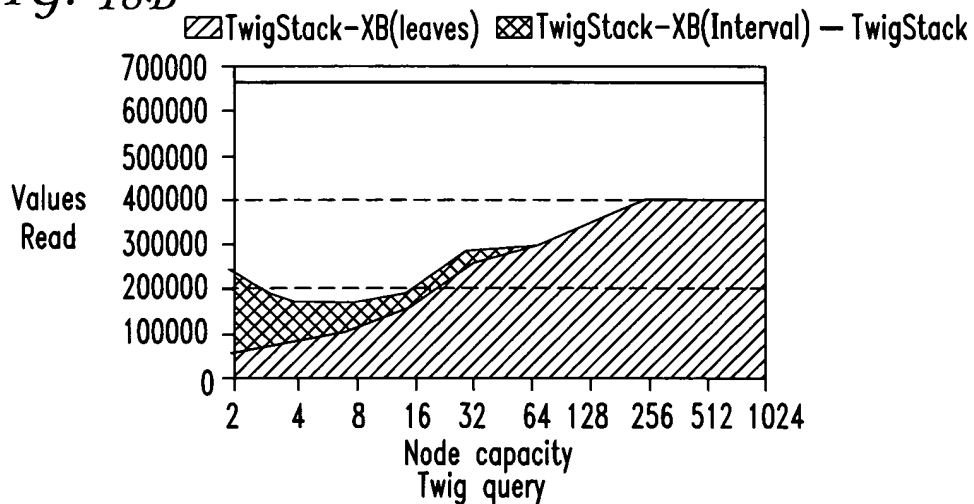


FIG. 18C

